Improved Dynamic Virtual Bats Algorithm for Global Numerical Optimization

Ali Osman Topal, Yunus Emre Yildiz, and Mukremin Ozkul

***Abstract*—Dynamic Virtual Bats Algorithm (DVBA) is a relatively new nature inspired optimization algorithm. DVBA, like Bat Algorithm (BA), is fundamentally inspired by bat's hunting strategies, but it is conceptually very different from BA. In DVBA, a role based search is developed to avoid deficiencies of BA. Although the new technique outperforms BA significantly, there is still an insufficiency in DVBA regarding its exploration, when it comes to high dimensional complex optimization problems. To increase the performance of DVBA, this paper presents a novel, improved dynamic virtual bats algorithm (IDVBA) based on probabilistic selection. The performance of the proposed IDVBA was compared with DVBA and four state-of-the-art BA variants. The algorithms were tested on 30 optimization problems from CEC 2014. The experimental results demonstrated that the new search mechanism improved the performance of DVBA in terms of accuracy and robustness.**

***Index Terms*—dynamic virtual bats algorithm, meta heuristic algorithm, global numerical optimization.**

I. Introduction

DYNAMIC virtual bat algorithm (DVBA), by Topal and Altun [13], [14], is another meta-heuristic algorithm inspired by bats ability to manipulate frequency and wavelength of emitted sound waves. DVBA is not a classic Bat Algorithm [17] variation, it is a new simulation of the bat's hunting strategies. In DVBA, a role-based search is developed by using just two bats to improve the Bat Algorithm. Recently, DVBA has been tested on well-known test functions [13] and supply chain cost problem [15]. Experimental results show that, DVBA is suitable for solving most of the low dimensional problems. However, DVBA, similar to other evolutionary algorithms, has some challenging problems. For example, the convergence speed of DVBA is slower than other population-based algorithms like PSO [9], GA [5], BF [12], and BA. The convergence issue of these algorithms was resolved through the latest variants [4] [19]. Additionally, in high dimensional multimodal problems, escaping from the local optima traps becomes a difficult task for DVBA. Therefore, accelerating convergence speed and avoiding the local optima have become two of the most important issues in DVBA.

To minimize the impact of this weakness, this paper proposes an improved solution which accelerates convergence speed and avoids the local optima trap. To achieve both goals, we introduce a new search mechanism for the explorer bat. This new search mechanism improves the search performance and gives DVBA more powerful exploitation capabilities. Simulations and comparisons based on several well-studied benchmarks demonstrate the effectiveness, efficiency, and robustness of the proposed IDVBA.

The rest of this paper is organized as follows. Section 2 summarizes DVBA. The improved DVBA algorithm is presented in Section 3. Section 4 presents simulation results on the use of IDVBA for solving CEC 2014 test functions. Finally, conclusions are drawn in Section 5.

Manuscript received July 7, 2017; revised July 28, 2017.

A.O.Topal is with the Department of Computer Engineering, Epoka University, Tirana, Albania e-mail: aotopal@epoka.edu.al.

Y.E. Yildiz is with the Department of Computer Engineering, Metrapolitan University, Tirana, Albania e-mail: theyeyildiz@gmail.com.

M. Ozkul is with the Department of Computer Engineering, Epoka University, Tirana, Albania e-mail: mozkul@epoka.edu.al.

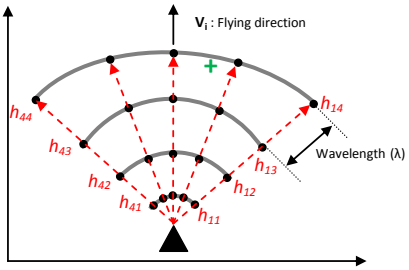

Fig. 1. Explorer bat is searching for prey with a wide search scope.

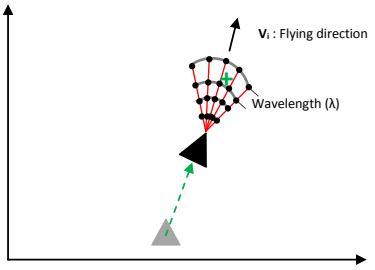

Fig. 2. Exploiter bat is chasing prey with a narrow search space.

II. Dynamic Virtual Bats Algorithm (DVBA)

When bats search out prey, they burst sound pulses with lower frequency and longer wavelengths so the sound pulses can travel farther. In this long range mode it becomes hard to detect the exact position of the prey; however, it becomes easy to search a large area. When bats detect prey, the pulses will be emitted with higher frequency and shorter wavelengths so that bats are able to update the prey location more often [1], [7]. In DVBA, two bats are used to imitate this hunting behavior. Each bat has its own role in the algorithm and during the search they exchange these roles according to their positions. These bats are referred as explorer bat and exploiter bat. The bat that is in a better position becomes the exploiter meanwhile the other becomes the explorer.

In Fig.1 and Fig.2 [13], the hunting strategy of a bat is simulated. The black triangle is the current solution (bat location), the black circles are the visited solutions on the search

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Algorithm 1 DVBA pseudo code. f_{gbest} is the global best solution and d is the number of dimensions. [13]

```
Objective function f(x), x = (x_1, ..., x_d)^T
Initialize the bat population x_i (i = 1, 2) and v_i
Initialize wavelength λ_i and frequency f_i
Initialize the number of the waves
while (t < Max number of iterations) do
    for each bat do
        Create a sound waves scope
        Evaluate the solutions on the waves
        Choose the best solution on the waves, h_*
    end for
    if (f(h_*) < f(x_i)) then
        Move to new solution
        Decrease λ_i and increase f_i
    else if (f(x_i) > f_gbest) then
        Change the direction randomly
        Increase λ_i and decrease f_i
    else if (f(x_i) = f_gbest) then
        Minimize λ_i and maximize f_i
        Change the direction randomly
    end if
    end if
    Rank the bats and find the current best x_gbest
end while
end while
```

waves in this iteration, and "+" sign represents prey. During the search, the explorer bat's search scope gets its widest shape; the distance between the search waves and the angle between the wave vectors (red dashed arrows) get larger (see Fig. 1). On the contrary, if the bat becomes the exploiter bat, its search scope gets its narrowest shape; the distance between the search waves and the angle between the wave vectors get smaller (see Fig.2). The number of the visited solutions is same for both bats, just the distance between the solutions changes dynamically by using wavelength and frequency. The wavelength and the distance between the solutions are proportional. The frequency and the angle between the wave vectors are inversely proportional.

The algorithm determines the best solution in the bat's search scope. If it is better than the current location (solution), the bat will fly to the better solution, decrease the wavelength and increase the frequency for the next iteration. These changes are targeting to increase the intensification of the search. Unless there is no better solution than the current solution in the search scope, the bat will stay on it, turn around randomly and keep scanning its nearby surrounding space. It will keep spinning in this position and expanding its search scope until it finds a better solution. The basic steps of the algorithm for minimizing an objective function $f(x)$ are shown in Algorithm 1.

III. Proposed Novel Improved Dynamic Virtual Bats Algorithm (IDVBA)

A. The weakness of DVBA

In DVBA, the explorer bat's search scope size is limited by the wavelength. This search scope might not be large enough to detect better solutions near its surrounding space. Thus, it is very likely that the explorer bat will be trapped in a local optimum. In addition, the exploiter bat's search scope size becomes very small during the exploitation and it moves very slowly. Therefore it might need too much time to reach the global optima. These problems in DVBA have been eradicated by introducing probabilistic selection restart techniques in IDVBA.

The concept of IDVBA is analogous to the idea behind the Micro-Particle Swarm Optimizer [6] and the Micro-Genetic Algorithm [10], where a set of restart operations are executed after the population has converged. However, IDVBA uses restart operations according to the bats' stagnancy during the search and it doesn't blacklist the inferior solutions as in micro-genetic algorithm.

B. Improved Dynamic Virtual Bats Algorithm (IDVBA)

To improve the search performance and give DVBA more powerful search capabilities, we introduce two probabilistic selections: *R* - random flying probability and *C* - convergence probability. If the explorer bat is stuck in large local minima, it chooses to fly away from the trap randomly with a probability *R* related to number of unsuccessful attempts. In addition, it chooses to fly near to the exploiter bat with a probability *C* related to the number of escapes attempted from the traps. *R* is calculated as follows:

$$R_i^{t+1} = R_i^t[1 - exp(-trial_i\gamma)], \tag{1}$$

where γ constant and $trial_i$ denotes the number of unsuccessful attempts. Obviously, the higher the $trial_i$ is, the greater the probability that the explorer bat might fly away from the trap to a random solution in the search space. As the unsuccessful attempts increase, the random flying probability R_i decreases and the possibility of $rand() < R_i$ being true

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Algorithm 2 IDVBA pseudo code. f_{gbest} is the global best solution and d is the number of dimensions. The code we discuss in the text is in boldface.

1: Objective function *f(x)*, $x = (x_1, ..., x_d)^T$
2: Initialize the bat population $x_i (i = 1, 2)$ and v_i
3: Initialize wavelength λ_i and frequency f_i
4: Initialize the number of the waves
5: $C_i = R_i = 1.0$
6: while ($t <$ Max number of iterations) do
7: for *each* bat do
8: Create a sound waves scope
9: Evaluate the solutions on the waves
10: Choose the best solution on the waves, h_*
11: end for
12: if $f(h_*) < f(x_i)$ then
13: Move to new solution
14: Decrease λ_i and increase f_i
15: $\boldsymbol{trial_i = 0}$
16: else if $f(x_i) > f_{gbest}$ then
17: ***Calculate the random flying probability R_i by Eq.1***
18: if $\boldsymbol{rand() \text{ ¡ } R_i}$ then
19: Change the direction randomly
20: $\boldsymbol{trial_i = trial_i + 1}$
21: else
22: ***Restart the search from a random position***
23: ***Reset R_i and $trial_i$***
24: $\boldsymbol{rfly_i = rfly_i + 1}$
25: ***Calculate the C_i by Eq.2***
26: end if
27: if $\boldsymbol{rand() > C_i}$ then
28: ***Produce a new solution around the exploiter bat.***
29: ***Reset C_i and $rfly_i$***
30: end if
31: Increase λ_i and decrease f_i
32: else if $f(x_i) = f_{gbest}$ then
33: Minimize λ_i and maximize f_i
34: Change the direction randomly
35: ***Reset $rfly_i$ and $trial_i$***
36: end if
37: end if
38: Rank the bats and find the current best x_{gbest}
39: end while
40: end while

(line 17 in Algorithm 2) increases. This can help the explorer bat to escape from the local optima trap rapidly. However, the explorer bat should not leave the trap without exploring the nearby surrounding space. Thus, γ should be chosen carefully.

The convergence probability C gives possibility to the explorer bat to converge with the exploiter bat, instead of exploring a random position. Thus, the exploitation speed will be increased rapidly around the best position. Time after time the explorer bat visits the exploiter bat to speed up the exploitation process then flies away randomly to keep up the exploration process. This also increases the exploitation capability of IDVBA. C is calculated as follows:

$$C_i^{t+1} = C_i^t[1 - exp(-rfly_i\gamma)], \quad (2)$$

where $rfly_i$ denotes the number of random restarts. As shown in Eq.(2), the convergence probability C is inversely proportional with the number of random restarts $rfly_i$ done by the explorer bat. Thus, increasing random restarts will increase the probability of $rand() > C$ that allows the explorer bat to visit the exploiter bat often (line 25-26 in Algorithm 2).

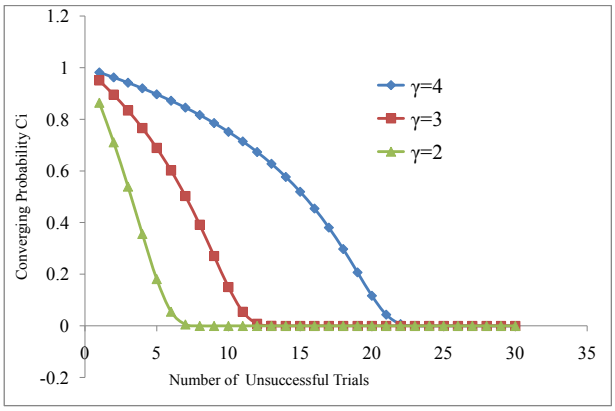

Fig. 3. Characteristics of Eq. 2 C - convergence probability changes as the unsuccessful trials increases for $\gamma = 2, 3$ and 4.

Fig.3 shows the characteristics of Eq. 2 for different values of γ as the unsuccessful attempts of the explorer bat increases. As it can be seen from Fig.3 and Algorithm 2, the higher γ value will keep the explorer bat longer in local minima. Our preliminary studies have suggested that the best value for γ is in the range of [2, 4] for the test

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

functions we use in this work. *R* is calculated with the same equation used for *C* Eq. 2 but with different values of γ. Therefore Fig.3 shows the characteristics of Eq.1 as well. By properly choosing the value of γ, we can prevent bats from local minima traps while allowing them to explore and exploit the nearby surrounding space without experiencing too many unnecessary random jumps. In our experiments, we set $\gamma = 2$ for *R* and $\gamma = 3$ for *C*. According to all these approximations and improvements IDVBA can be given as in Algorithm 2. The differences from DVBA are shown in boldface font.

IV. Experiments

This section presents an extensive comparisons among the performances of six algorithms. IDVBA is compared with DVBA and four state-of-the-art BA variants: Adaptive Bat Algorithm (ABA) [16], Novel Adaptive Bat Algorithm (NABA) [8], Local Memory Search Bat Algorithm (LMSBA) [18], and Chaotic Local Search-based Bat Algorithm (CLSBA) [2].

In our experimental studies, parameter settings of the algorithms are same as in their original papers. The population size of the algorithms has been kept as 50.

All the algorithms are developed using Anaconda Python Distribution environment and run a PC with a 3.0 GHz CPU and 8.00 GB of RAM.

A. Test Functions

To provide a comprehensive comparison, all thirty CEC 2014 test functions are used to conduct this experiment. The descriptions of these functions are given in [11]. In CEC 2014 test suits, $f_1 - f_3$ are rotated unimodal, $f_4 - f_{16}$ are shifted and rotated multimodal, $f_{17} - f_{22}$ are hybrid, and $f_{23} - f_{30}$ are composition test functions.

B. Comparison Experiments

This paper aims to test the quality of the final solution and the convergence speed at the end of a fixed number of function evaluations (FEs). According to the instructions in CEC 2014 special session we set the maximum number of FEs as 3×10^5 and the dimensions of the problems 30-D. The test results are shown in Table 1 in terms of the mean error values (MeanErr) and the standard deviation (STDEV) of the results found over the 30 independent runs by each algorithm.

Furthermore, we used t-tests [3] to compare the mean error values of the results produced by the IDVBA and the other algorithms at the 0.05 level of significance. The statistical significance level of the results are shown in the Table 1. In Table 1 '$-$' indicates that IDVBA is significantly more successful than selected one at a 0.05 level of significance by two-tailed test, '$\approx$' means the difference of means is not statistically significant and '$+$' indicates that the other algorithm is significantly more successful than IDVBA at a 0.05 level of significance by two-tailed test.

C. Experimental Results and Discussion

The functions f_1, f_2, and f_3 are unimodal and non-separable plate shaped problems. From Table 1, it can be seen that these functions are very hard to optimize and the algorithms did not show a significant success. By analyzing their t-test values, we can see that the solutions with IDVBA are significantly better than that with other algorithms, followed by NABA. For the function f_3, except DVBA, IDVBA is outperformed by other algorithms. NABA has the best solutions in terms of mean error values, followed by CLSBA.

The test functions in second group (f_4 - f_{16}) are simple multi-modal. The t-test values show that the performance of IDVBA is significantly better than that of other five algorithms. In this group, DVBA remained the toughest competitors of IDVBA in most of the functions.

The third group ($f_{17} - f_{22}$) has six hybrid functions which are almost same as real-world optimization problems [11]. On these functions, IDVBA exhibits better performance than other algorithms. For function f_{20}, LMSBA, NABA, and ABA have less mean error values than that of IDVBA. LMSBA, CLSBA, and NABA performed comparable to IDVBA on f_{19}. In this group, LMSBA and NABA also find competitive solutions compared with IDVBA, as their t-test values reflect.

In group four, there are eight composition functions. The composition function merges the properties of the sub-functions better and maintains continuity around the global/local optima. We can observe that, the performance of IDVBA is superior overall to that of five competitors. Only for function f_{24}, LMSBA performed better than IDVBA in terms of the mean error values.

We observe that IDVBA found effective results in most cases, which implies that IDVBA by its improved search mechanism is more competent in tackling complex problems than DVBA.

Additionally, we can observe that among the BA's variants, NABA has the best performance in terms of the mean, followed by LMSBA.

V. Conclusion

In order to apply DVBA to solve complex high dimensional problems efficiently, this paper proposes a novel improved dynamic virtual bats algorithm, namely IDVBA. The proposed algorithm employs two probabilistic selections - random flying probability *R* and convergence probability *C*. They are used to prevent the bats from falling into the local minimum traps, accelerate the convergence speed, and increase the accuracy. Results show that we achieved our goal efficiently with the new search mechanism.

To prove the effectiveness and robustness of the proposed algorithms, the proposed IDVBA was compared with the DVBA, ABA, CLSBA, LMSBA, and NABA on all 30 bound-constrained numerical 30-D optimization problems from CEC-2014. The results demonstrate that the IDVBA achieves a good balance between exploration and exploitation and has the best universality on different type of problems. And we can say that IDVBA in general performs better or comparable to other algorithms.

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I
COMPARISON OF LMSBA, NABA, CLSBA, ABA, DVBA, AND IDVBA ON ALL 30 TEST FUNCTIONS OF CEC 2014.

Func.	LMSBA		NABA		CLSBA		ABA		DVBA		IDVBA
	MeanErr (StdDev)	t-test	MeanErr (StdDev)	t-test	MeanErr (StdDev)	t-test	MeanErr (StdDev)	t-test	MeanErr (StdDev)	t-test	MeanErr (StdDev)
f_1	6.25E+6(2.25E+6)	$-$	6.18E+6(1.19E+6)	$-$	6.38E+6(1.22E+6)	$-$	7.33E+6(2.21E+6)	$-$	1.08E+5(4.30E+5)	$-$	6.65E+4 (1.20E+4)
f_2	7.53E+7(2.61E+6)	$-$	5.46E+7(2.92E+6)	$-$	6.47E+7(6.56E+6)	$-$	7.18E+7(4.93E+6)	$-$	2.82E+4(1.29E+4)	$+$	2.31E+5 (2.11E+4)
f_3	1.36E+3(5.39E+2)	$+$	2.48E+2(2.73E+1)	$+$	5.20E+2(1.02E+2)	$+$	1.54E+3(8.97E+2)	$+$	1.58E+4(5.78E+3)	$-$	9.19E+3 (3.21E+3)
f_4	1.02E+2(4.45E+1)	$\approx$	1.26E+2(3.98E+1)	$-$	1.70E+2(7.83E+1)	$-$	1.29E+2(3.40E+1)	$-$	1.08E+2(3.34E+1)	$\approx$	9.51E+1 (1.23E+1)
f_5	2.08E+1(3.92E-2)	$-$	2.11E+1(3.08E-2)	$-$	2.12E+1(3.13E-2)	$-$	2.09E+1(3.07E-2)	$-$	2.00E+1(5.92E-2)	$+$	2.03E+1 (2.16E-2)
f_6	2.79E+1(4.52E+0)	$-$	2.93E+1(3.11E+0)	$-$	2.93E+1(4.01E+0)	$-$	2.90E+1(1.14E+0)	$-$	2.32E+1(3.18E+0)	$-$	5.48E+0 (9.96E-1)
f_7	1.58E+0(9.03E-2)	$-$	1.48E+0(4.48E-2)	$-$	1.56E+0(9.90E-2)	$-$	1.59E+0(6.72E-2)	$-$	1.09E+0(1.13E-2)	$-$	1.06E+0 (7.53E-3)
f_8	2.32E-1(2.82E-1)	$-$	2.32E-1(2.43E-1)	$-$	2.49E-1(5.52E-1)	$-$	3.18E-1(4.87E+1)	$-$	3.11E-2(2.01E-1)	$\approx$	2.37E-3 (8.42E-4)
f_9	5.67E+1(4.97E+1)	$-$	5.83E+1(5.87E+1)	$-$	4.60E+01(3.03E+1)	$\approx$	6.00E+1(5.32E+1)	$-$	2.13E+1(9.60E+0)	$+$	3.86E+1 (3.93E+0)
f_{10}	1.51E+3(6.57E+2)	$-$	1.05E+3(6.02E+2)	$-$	1.69E+3(6.34E+2)	$-$	1.73E+3(5.72E+2)	$+$	8.02E+2(1.54E+2)	$-$	6.16E+2 (1.28E+2)
f_{11}	5.04E+3(3.78E+2)	$-$	5.26E+3(3.57E+2)	$-$	5.14E+3(6.10E+2)	$-$	5.20E+3(6.37E+2)	$-$	3.93E+3(5.23E+2)	$-$	6.67E+2 (3.47E+2)
f_{12}	2.11E+0(1.99E-1)	$-$	1.93E+0(1.54E-1)	$-$	2.03E+0(1.94E-1)	$-$	1.87E+0(3.16E-1)	$+$	9.80E-1(3.07E-1)	$\approx$	9.43E-1 (1.36E-1)
f_{13}	9.87E-1(3.05E-2)	$-$	4.90E-1(1.27E-1)	$\approx$	4.90E-1(7.33E-2)	$-$	4.30E-1(6.11E-2)	$\approx$	5.10E-1(1.03E-1)	$-$	4.34E-1 (8.54E-2)
f_{14}	3.10E-1(3.48E-2)	$\approx$	2.60E-1(3.74E-2)	$\approx$	2.70E-1(5.46E-2)	$\approx$	2.90E-1(4.71E-2)	$-$	2.00E-1(2.05E-2)	$\approx$	2.30E-1 (2.14E-1)
f_{15}	2.14E+1(1.58E+0)	$-$	5.87E+0(1.26E+0)	$-$	1.86E+1(1.16E+0)	$-$	2.11E+01(6.11E-1)	$-$	2.49E+0(6.33E-1)	$\approx$	2.22E+0 (1.75E-1)
f_{16}	1.41E+1(1.76E-1)	$-$	1.33E+1(4.86E-1)	$-$	1.34E+1(3.58E-1)	$-$	1.30E+1(7.35E-1)	$-$	1.27E+1(4.27E-1)	$-$	1.22E+1 (5.74E-1)
f_{17}	1.02E+5(3.27E+5)	$-$	1.26E+5(6.67E+4)	$-$	1.29E+5(1.84E+5)	$-$	1.70E+5(3.09E+5)	$-$	8.09E+4(3.42E+4)	$-$	8.39E+3 (4.67E+3)
f_{18}	9.91E+4(2.77E+5)	$-$	4.15E+5(2.9E+5)	$-$	1.04E+5(1.17E+4)	$-$	2.17E+5(1.36E+5)	$-$	7.90E+3(9.35E+3)	$-$	8.25E+2 (7.28E+2)
f_{19}	1.47E+1(1.78E+0)	$\approx$	1.41E+01(1.97E+0)	$+$	1.49E+1(7.02E-1)	$\approx$	2.80E+1(2.71E+1)	$-$	1.81E+1(2.33E+0)	$-$	1.50E+1 (1.28E+0)
f_{20}	3.52E+2(5.14E+1)	$+$	3.87E+2(1.16E+2)	$+$	5.30E+2(5.34E+1)	$\approx$	4.06E+2(1.07E+2)	$+$	8.29E+2(3.22E+2)	$-$	5.51E+2 (1.44E+2)
f_{21}	8.41E+4(6.62E+4)	$\approx$	7.57E+4(3.51E+4)	$\approx$	1.13E+5(6.87E+4)	$-$	1.36E+5(8.29E+4)	$-$	8.45E+4(4.28E+4)	$\approx$	9.40E+4 (4.70E+4)
f_{22}	8.97E+2(1.67E+2)	$-$	1.00E+3(2.59E+2)	$-$	1.03E+3(8.22E+1)	$-$	1.01E+3(2.06E+2)	$-$	4.70E+2(1.76E+2)	$\approx$	3.70E+2 (1.63E+2)
f_{23}	3.21E+2(4.98E-1)	$-$	3.17E+2(3.63E-1)	$-$	3.17E+2(3.74E-1)	$-$	3.19E+2(2.49E+0)	$-$	3.16E+2(3.61E-1)	$\approx$	3.16E+2 (6.31E-2)
f_{24}	2.33E+2(6.72E+0)	$-$	2.43E+2(1.31E+1)	$-$	2.55E+2(3.28E+1)	$-$	2.39E+2(7.93E+0)	$-$	2.49E+2(1.04E+1)	$-$	1.40E+2 (7.43E+0)
f_{25}	2.33E+2(1.10E+1)	$-$	2.17E+2(1.96E+1)	$-$	2.19E+2(1.13E+1)	$-$	2.29E+2(3.73E+1)	$-$	2.19E+2(9.11E+0)	$-$	2.11E+2 (2.46E+0)
f_{26}	1.03E+2(5.83E-2)	$-$	1.20E+2(4.01E+1)	$-$	1.41E+2(4.90E+1)	$-$	1.00E+2(9.34E-2)	$\approx$	1.00E+2(7.53E-2)	$\approx$	1.00E+2 (4.73E-2)
f_{27}	5.47E+2(2.77E+2)	$\approx$	1.35E+3(9.83E+1)	$-$	1.25E+3(9.83E+1)	$-$	1.06E+3(3.40E+2)	$-$	5.70E+2(2.61E+2)	$\approx$	6.50E+2 (3.09E+2)
f_{28}	2.13E+3(7.81E+2)	$-$	2.52E+3(6.41E+2)	$-$	3.78E+3(3.81E+2)	$-$	2.50E+3(1.12E+2)	$-$	1.56E+3(3.08E+2)	$-$	1.27E+3 (1.49E+2)
f_{29}	1.79E+6(5.14E+6)	$-$	6.71E+6(9.27E+6)	$-$	1.87E+6(3.63E+6)	$-$	4.06E+6(5.06E+6)	$-$	1.84E+4(2.20E+4)	$-$	1.78E+4 (8.01E+3)
f_{30}	4.47E+3(6.03E+2)	$+$	1.11E+4(4.67E+3)	$\approx$	1.19E+4(8.17E+3)	$-$	1.48E+4(1.73E+3)	$-$	1.19E+4(5.09E+3)	$-$	9.20E+3 (3.78E+3)
$-$	22		22		23		23		16		
$+$	4		3		1		5		3		
$\approx$	4		5		6		2		11		

"+", "−", and "≈" denote that the performance of the corresponding algorithm is significantly better than, worse than, and significantly not different to that of DVBA, respectively.

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

REFERENCES

[1] M. Airas. Echolocation in bats. In *Proceedings of spatial sound perception and reproduction. The postgrad seminar course of HUT Acoustics Laboratory*, pages 1–25, 2003.

[2] C. Chandrasekar et al. An optimized approach of modified bat algorithm to record deduplication. *International Journal of Computer Applications*, 62(1), 2013.

[3] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar. Differential evolution using a neighborhood-based mutation operator. *Evolutionary Computation, IEEE Transactions on*, 13(3):526–553, 2009.

[4] D. Handayani, N. Nuraini, O. Tse, R. Saragih, and J. Naiborhu. Convergence analysis of particle swarm optimization (pso) method on the with-in host dengue infection treatment model. In *AIP Conference Proceedings*, volume 1723, page 030013. AIP Publishing, 2016.

[5] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press, 1975.

[6] T. Huang and A. S. Mohan. Micro-particle swarm optimizer for solving high dimensional optimization problems (μpso for high dimensional optimization problems). *Applied Mathematics and Computation*, 181(2):1148–1154, 2006.

[7] L. Jakobsen and A. Surlykke. Vespertilionid bats control the width of their biosonar sound beam dynamically during prey pursuit. *Proceedings of the National Academy of Sciences*, 107(31):13930–13935, 2010.

[8] M. W. U. Kabir, N. Sakib, S. M. R. Chowdhury, and M. S. Alam. A novel adaptive bat algorithm to control explorations and exploitations for continuous optimization problems. *International Journal of Computer Applications*, 94(13), 2014.

[9] J. Kennedy. Eberhart c., iparticle swarm optimization,". In *Proceedings of IEEE International Conference on Neural Network*, pages 1942Y1948–1995, 1995.

[10] M. Koppen, K. Franke, and R. Vicente-Garcia. Tiny gas for image processing applications. *Computational Intelligence Magazine, IEEE*, 1(2):17–26, 2006.

[11] J. Liang, B. Qu, and P. Suganthan. Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory*, 2013.

[12] K. M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *Control Systems, IEEE*, 22(3):52–67, 2002.

[13] A. O. Topal and O. Altun. Dynamic virtual bats algorithm (dvba) for global numerical optimization. In *Intelligent Networking and Collaborative Systems (INCoS), 2014 International Conference on*, pages 320–327. IEEE, 2014.

[14] A. O. Topal and O. Altun. A novel meta-heuristic algorithm: Dynamic virtual bats algorithm. *Information Sciences*, 354:222–235, 2016.

[15] A. O. Topal, O. Altun, and E. Terolli. Dynamic virtual bats algorithm (dvba) for minimization of supply chain cost with embedded risk. In *Proceedings of the 2014 European Modelling Symposium*, pages 58–64. IEEE Computer Society, 2014.

[16] X. Wang, W. Wang, and Y. Wang. An adaptive bat algorithm. In *Intelligent Computing Theories and Technology*, pages 216–223. Springer, 2013.

[17] X.-S. Yang. A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74. Springer, 2010.

[18] M. Yuanbin, Z. Xinquan, and X. Shujian. Local memory search bat algorithm for grey economic dynamic system. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 11(9):4925–4934, 2013.

[19] Y. E. Yldz and O. Altun. Hybrid achievement oriented computational chemotaxis in bacterial foraging optimization: a comparative study on numerical benchmark. *Soft Computing*, pages 1–17, May 2015.

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)